

---

# 基于组合范畴语法的中文语义分析 说明文档

章晔  
2016/5/2

---

# 目录

一、组合范畴语法.....	3
1.1 词典 .....	3
1.2 语法规则 .....	3
二、逻辑表达式 .....	4
2.1 Lambda Calculus.....	4
三、模型设计 .....	5
3.1 对数线性模型 .....	5
3.2 特征.....	6
3.3 参数估计 .....	6
四、词典生成.....	7
4.1 词典初始化.....	7
4.2 新词条生成.....	8
4.3 词条拆分 .....	8
五、数据标注 .....	10
5.1 Lambda Calculus.....	10
5.2 标注方案 .....	11
5.3 数据说明 .....	12
5.4 数据文件 .....	13
六、实验 .....	13
七、参考文献.....	13

---

# 一、组合范畴语法

组合范畴语法（Combinatory Categorical Grammar）是一种将句法（syntax）和语义（semantics）紧密结合的语言学表示，可以用来模拟广泛的语言现象。用于语义分析的组合范畴语法主要包含词典和组合规则。

## 1.1 词典

词典中每个词条以如下形式表示：

**words** :- **X** : **h**

其中 **words** 表示自然语言中的短语或句子，**X** 表示该词组的句法成分，**h** 是一个逻辑表达式，表示该短语对应的语义。

句法成分中，**S** 表示句子，**NP** 表示名词成分。在本项目中，所有的句法成分都用 **S** 和 **NP** 的组合表示。比如 **S\NP** 表示该短语可以与前面的名词成分 **NP** 构成一个句子 **S**，**S/NP** 表示该短语可以与后面的名字成分 **NP** 构成一个句子 **S**。**S|NP** 表示 **S\NP** 或者 **S/NP** 均可，例如 **S/(S|NP)** 后面接 **S/NP**、**S\NP** 或 **S|NP** 都可以构成一个句子 **S**。

逻辑表达式用 **Lambda Calculus** 表示。

词条示例：

西单 :- **NP** : 西单:s

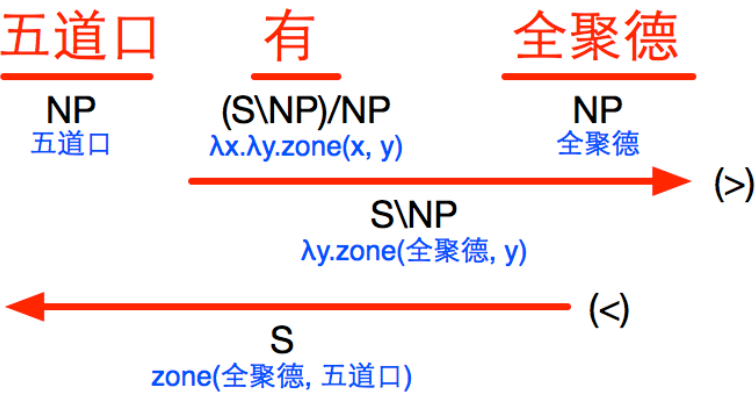
附近有 :- **S\NP/NP** :  $\lambda x \lambda y. zone(x, y)$

## 1.2 语法规则

组合范畴语法根据一些组合规则，将相邻的短语连接起来，形成组合的句法成分和语义。本项目中用到的是四条基本规则：

$X/Y:f \ Y/Z:g \Rightarrow X/Z:\lambda x.f(g(x))$	forward composition (>B)
$X/Y:f \ Y:g \Rightarrow X:f(g)$	forward apply (>)
$Y/Z:g \ X\Y:f \Rightarrow X\Z:\lambda x.f(g(x))$	backward composition (<B)
$Y:g \ X\Y:f \Rightarrow X:f(g)$	backward apply (<)

比如对于“五道口有全聚德”这句话，由三个短语“五道口”“有”“全聚德”构成，对相邻的短语根据规则进行组合，最后得到整个句子的句法成分是 S，表示这是一句完整的语句，并且解析出这句话的语义。



## 二、逻辑表达式

语义分析（Semantic Parsing）是将自然语句转化成逻辑表达式来表示语义。本项目逻辑表达式使用 Lambda Calculus。

### 2.1 Lambda Calculus

Lambda Calculus 有三种类型：t 表示真值类型，e 表示实体类型，i 表示数字。Lambda Calculus 中也有函数，即表示实体之间的关系。函数类型由输入类型和返回类型组成，比如  $\text{price}(x)$  的类型是  $\langle e, i \rangle$ ，输入  $x$  是实体，返回值是数字； $\text{restaurant}(x)$  的类型是  $\langle e, t \rangle$ ，输入是实体，返回值是真值。

其中实体 e，还可以根据特定领域设定子类型。

Lambda Calculus 表达式有以下几种部分构成：

常量，又分为名词常量和关系常量。名词常量，比如“中关村”“火锅”这些地名或者标签，名词常量都是实体，需要标明实体类型。关系常量，即根据需要自定义的关系函数，比如商圈，人均价格。关系常量也需要标明函数类型。

逻辑联结词，即与、或、非。

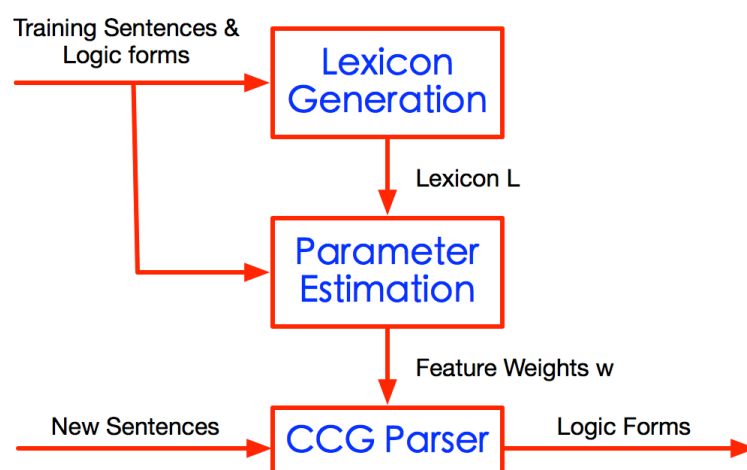
Lambda 项，表示使得后面表达式为真的项，比如  $\lambda x.\text{zone}(x, \text{五道口})$  表示存在  $(x, \text{zone}, \text{五道口})$  这种关系的实体。

逻辑表达式示例：

$\lambda x.\text{restaurant}(x) \wedge \text{zone}(x, \text{五道口}) \wedge \text{hasCuisine}(x, \text{麻辣小龙虾})$ ，表示五道口卖麻辣小龙虾的餐厅。

## 三、模型设计

利用组合范畴语法进行语义分析的过程如下图所示。



利用训练数据，即自然语句和对应的逻辑表达式，生成词典，并利用词典和训练数据进行参数估计，进行数轮迭代，最终得到一个组合范畴语法的语义分析器，包含一个词典和特征权重。

对于新的句子，通过语义分析器分析，可以得到对应的逻辑表达式。

### 3.1 对数线性模型

给定一个词典  $\Lambda$  和语法规则，对一个句子的语义分析可能产生很多结果。我们使用对数线性模型来选择可能性最高的结果。

给出一个句子  $x$ ，对这个句子进行语义分析，经过解析过程  $y$  得到逻辑表达式  $z$  的概率为：

$$P(y, z|x; \theta, \Lambda) = \frac{e^{\theta \cdot \phi(x, y, z)}}{\sum_{(y', z')} e^{\theta \cdot \phi(x, y', z')}}$$

其中， $\phi$  是特征向量， $\theta$  是参数向量。

对于句子  $x$ ，找出可能性最高的逻辑表达式，即：

$$f(x) = \arg \max_z p(z|x; \theta, \Lambda)$$

其中，逻辑表达式  $z$  的概率为所有得到此表达式的分析过程的概率之和：

$$p(z|x; \theta, \Lambda) = \sum_y p(y, z|x; \theta, \Lambda)$$

## 3.2 特征

给出句子  $x$ ，经过语义分析过程  $y$  得到逻辑表达式  $z$ ，特征向量为使用了词典中哪些词条。参数向量为词典中每一个词条的权重。

## 3.3 参数估计

我们使用随机梯度下降法进行参数估计，对每个样本更新一次。

给定  $n$  对训练数据  $(x_i, z_i)$ ， $i=1 \dots n$ 。每轮迭代，对于每对训练数据，都更新参数。

每个样本的最大化目标函数为：

$$O_i = \log P(z_i|x_i; \theta, \Lambda)$$

对  $\theta_j$  求偏导计算梯度  $\Delta$ ，计算公式为：

$$\begin{aligned} \frac{\partial O_i}{\partial \theta_j} &= E_{p(y|x_i, z_i; \theta, \Lambda)} [\phi_j(x_i, y, z_i)] \\ &\quad - E_{p(y, z|x_i; \theta, \Lambda)} [\phi_j(x_i, y, z)] \end{aligned}$$

更新参数：

$$\theta = \theta + \alpha \Delta$$

$\alpha$  为学习速率，随迭代增加而减小。

---

## 四、词典生成

首先，初始化词典。

每一轮迭代中，对于每一条样本的自然语句，根据当前词典进行语义分析，得到当前得分最高的一个解析树，从这条解析树拆分出新的词条加入词典，然后更新参数。

最终得到一个词典和词典中每个词条的权重。

### Initialization:

- Set  $\Lambda = \{x_i \vdash S : z_i\}$  for all  $i = 1 \dots n$ .
- Set  $\Lambda = \Lambda \cup \Lambda_{NP}$
- Initialize  $\theta$  using cooccurrence statistics, as described in Section 7.

### Algorithm:

For  $t = 1 \dots T, i = 1 \dots n$ :

#### Step 1: (Update Lexicon)

- Let  $y^* = \arg \max_y p(y|x_i, z_i; \theta, \Lambda)$
- Set  $\Lambda = \Lambda \cup \text{NEW-LEX}(y^*)$  and expand the parameter vector  $\theta$  to contain entries for the new lexical items, as described in Section 7.

#### Step 2: (Update Parameters)

- Let  $\gamma = \frac{\alpha_0}{1+c \times k}$  where  $k = i + t \times n$ .
- Let  $\Delta = E_{p(y|x_i, z_i; \theta, \Lambda)}[\phi(x_i, y, z_i)] - E_{p(y, z|x_i; \theta, \Lambda)}[\phi(x_i, y, z)]$
- Set  $\theta = \theta + \gamma \Delta$

**Output:** Lexicon  $\Lambda$  and parameters  $\theta$ .

### 4.1 词典初始化

将训练数据中每一对样本  $\langle x_i, z_i \rangle$  以词条的形式加入词典：

$$x_i :- S : z_i$$

然后，将固有名词加入词典，例如：

五道口 :- NP : 五道口:s

---

全聚德 :- NP : 全聚德:r

这些名词词条将有助于句子中名词实体的识别。

初始化词典中的词条权重初始值都为 10。

由于中文不像英语一样有分词，需要对中文进行分词，将一句话或短语变成一串 token，以便后面进行拆分。

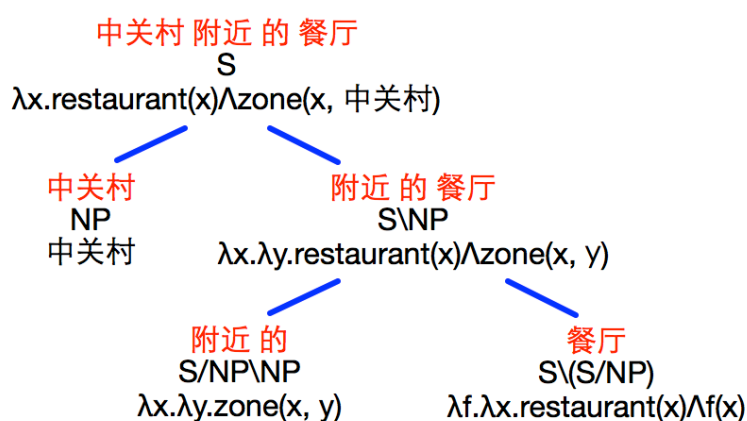
## 4.2 新词条生成

训练过程中，每轮迭代，对于每一个样本，根据已有词典，对自然语句  $x$  进行语义分析，产生当前得分最高的一个解析树  $y^*$ 。

$y^*$  中的叶节点即语义分析过程中用到的词条，而非叶结点是过程中产生的新词条。将非叶结点加入词典，并考虑其他可能产生此节点的解析：对非叶非根结点进行拆分，将得到的新词条也加入词典。

以下图“中关村附近的餐厅”一种解析树为例，利用“中关村”、“附近的”和“餐厅”三个词条，解析得到逻辑表达式。

解析树中有非叶非根节点“附近的餐厅”，将此词条加入词典，并对这个词条进行拆分，生成新的词条，这些新的词条都是可以产生“附近的餐厅”的词条，将这些新词条加入词典，并扩展特征向量。



## 4.3 词条拆分

语句是由短语构成的，组合范畴语法的本质就是根据词典理解每个短语的语义，并将短语组合的句子语义解析出来。



---

训练数据中的逻辑表达式是对一句完整的语句进行语义标注的，但是为了能够解析更多灵活变化的句子，需要得到每个短语的语义。这就是为什么要对解析树的非叶非根节点进行拆分得到新词条。

### 4.3.1 拆分限制

将逻辑表达式  $h$  拆分成  $f$  和  $g$ ， $f$  和  $g$  可以根据语法规则这，使得  $f(g)=h$  或者  $\lambda x.f(g(x))=h$ 。

如果不对  $f$  和  $g$  加以限制，可能会生成无数种可能性。例如：

$h=五道口$ ， $f=\lambda x.五道口$ ， $g$  可以是任何实体或关系，都能使得  $f(g)=h$ 。

再例如：

$h=f_1 \wedge f_2 \wedge f_3 \wedge f_4$ ， $h$  的任意子条件联结组合都可以分配给  $f$  或  $g$ 。

为了限制拆分可能性的数量，对拆分词条作如下限值：

#### 1、禁止无意义变量

$f$  和  $g$  都不能是  $\lambda x.e$  的形式，其中  $e$  是与变量  $x$  无关的逻辑表达式。

#### 2、 $h=f_1 \wedge f_2 \wedge f_3 \wedge f_4 \wedge \dots$ 时， $g$ 不能包含超过 $N$ 个子条件。本项目中 $N$ 为 4。

#### 3、 $f$ 不能是 $f=\lambda q.q(A)$ 的形式，其中 $q$ 是 $h$ 中不存在的变量且 $q$ 的输入是非变量。

### 4.3.2 逻辑表达式句法成分拆分

逻辑表达式  $h$  的句法成分为  $X$ ，即  $X:h$ ，将  $h$  拆分为  $f$  和  $g$ ， $f$  和  $g$  的句法成分取决于他们的类型。

以  $g$  为例， $g$  的类型为  $T(g)$ ，比如  $T(五道口)=e$ ， $T(\lambda x.zone(x, 五道口))=\langle e, t \rangle$ 。类型  $T$  的句法成分为  $C(T)$ ：

$$C(T) = \begin{cases} NP & \text{if } T = e \\ S & \text{if } T = t \\ C(T_2)|C(T_1) & \text{when } T = \langle T_1, T_2 \rangle \end{cases}$$

基本类型  $e$  的句法成分为名词成分  $NP$ ，基本类型  $t$  的句法成分为句子  $S$ ，所有其他类型的句法成分可以递归得到。例如  $C(\langle e, t \rangle)=S|NP$ ， $C(\langle e, \langle e, t \rangle \rangle)=S|NP|NP$ 。

分别对应四条语法规则：

---

$X/Y:f \ Y/Z:g \Rightarrow X/Z:\lambda x.f(g(x))$	forward composition (FC)
$X/Y:f \ Y:g \Rightarrow X:f(g)$	forward apply (FA)
$Y/Z:g \ X\backslash Y:f \Rightarrow X\backslash Z:\lambda x.f(g(x))$	backward composition (BC)
$Y:g \ X\backslash Y:f \Rightarrow X:f(g)$	backward apply (BA)

以四种方式拆分逻辑表达式和句法成分：

$$FA(X:h) = \{(X/Y:f, Y:g) \mid h=f(g) \wedge Y=C(T(g))\}$$

$$BA(X:h) = \{(Y:g, X\backslash Y:f) \mid h=f(g) \wedge Y=C(T(g))\}$$

$$FC(X/Y:h) = \{(X/W:f, W/Y:g) \mid h=\lambda x.f(g(x)) \wedge W=C(T(g(x)))\}$$

$$BC(X\backslash Y:h) = \{(W\backslash Y:g, X\backslash W:f) \mid h=\lambda x.f(g(x)) \wedge W=C(T(g(x)))\}$$

对  $A = X:h$  所有可能拆分  $S_C(A)$ :

$$S_C(A) = \{FA(A) \cup BA(A) \cup FC(A) \cup BC(A)\}$$

### 4.3.3 词条拆分

将逻辑表达式  $h$  和句法成分  $X$  拆分后，对短语（或句子） $w$  进行拆分。

已分词的  $w$  由  $n$  个 token 组成， $w_{0:n} = \langle w_0, w_1, \dots, w_n \rangle$ 。

对词条  $w_{0:n} :- A$  的拆分  $S(w_{0:n} :- A)$ :

$$S_L(w_{0:n} :- A) = \{(w_{0:i} :- B, w_{i+1:n} :- C) \mid 0 \leq i < n \wedge (B, C) \in S_C(A)\}$$

即枚举短语  $w$  所有拆分方式，与逻辑表达式和句法所有可能拆分方式组合。

## 五、数据标注

### 5.1 Lambda Calculus

Lambda Calculus 有三种类型： $t$  表示真值类型， $e$  表示实体类型， $i$  表示数字。其中实体  $e$ ，还可以根据特定领域设定子类型。

根据 RDF 知识库的设计方案，实体类型  $e$  的子类型有：

$r$ ，表示餐馆；

$c$ ，表示菜品；

---

s，表示字符串，餐馆和菜品的各种属性值类型都是 s。

Lambda Calculus 中的函数，即表示实体之间的关系，类型由输入类型和返回类型组成，比如  $\text{price}(x)$  的类型是  $\langle r, i \rangle$ ，输入  $x$  是餐馆，返回值是数字； $\text{restaurant}(x)$  的类型是  $\langle r, t \rangle$ ，输入  $x$  是餐馆，返回值是真值。

Lambda Calculus 表达式有以下几种部分构成：

- 常量，又分为名词常量和关系常量。名词常量，比如“中关村:s”、“火锅:s”、“全聚德:r”，这些地名或者标签，名词常量都是实体，需要标明实体类型。关系常量，即根据需要自定义的关系函数，比如商圈，人均价格。关系常量也需要标明函数类型。
- 逻辑联结词，即与、或、非，用 **and**，**or** 和 **not** 表示。
- Lambda 项， $\lambda x.e$  表示使得后面表达式  $e$  为真的  $x$ 。比如  $\lambda x.\text{zone}(x, \text{五道口})$  表示存在  $(x, \text{zone}, \text{五道口})$  这种关系的实体。  
若  $x$  是实体变量， $\lambda x$  用 `lambda $0 e` 表示；  
若  $x$  是关系变量，比如类型为  $\langle e, t \rangle$ ， $\lambda x$  用 `lambda $0 <e, t>` 表示。

不过标注数据时不会出现这种情况，一般出现在语义分析器拆分得到的词条中。

$\lambda x.\text{zone}(x, \text{五道口}) \rightarrow (\text{lambda } \$0 e (\text{zone:t } \$0 \text{ 五道口:s}))$

## 5.2 标注方案

用于初始化词典的 `np_lexicon` 中应有知识库中提取出的所有的名词常量，如行政区，商圈，餐厅种类，以及餐厅名和菜名等，例如：

五道口 :- NP : 五道口:s

全聚德 :- NP : 全聚德:r

北京烤鸭 :- NP : 北京烤鸭:c

知识库中餐厅名有些是带有“（XX 店）”的，应提取餐厅名字，忽略 XX 店。

根据数据，定义以下关系：

---

以 (关系名:resType argType1 argType2 ... resType) 的形式表示, arg Type 表示参数类型, resType 表示返回类型。

(restaurant:t r t), 表示 r 是否为餐厅。参数类型是 r, 返回类型是 t。

(name:t r s t), 表示 r 的名字是 s。

(district:t r s t), 表示 r 的行政区是 s。

(zone:t r s t), 表示 r 的商圈是 s。

(street:t r s t), 表示 r 的街道是 s。

(lat:i r i), r 的纬度。

(lng:i r i), r 的经度。

(price:i r i), r 的人均价格。

(address:s r s), r 的地址。

(tel:s r s), r 的电话。

(time:s r s), r 的营业时间。

(label:t r s t), r 的种类 (粤菜、火锅等等)

(tasteScore:i r i), r 的口味评分。

(surroundingScore:i r i), r 的环境评分。

(serviceScore:i r i), r 的服务评分。

(shopScore:i r i), r 的总体评分。

(hasCuisine:t r c t), 表示 r 有 c 这道菜。

(cuisinePrice:i c i), c 的价格。

(npeople:t r i t), 表示 r 有容纳 i 人的包间。

## 5.3 数据说明

标注了 407 条餐饮领域的问题, 每一个样本为<自然语句, 逻辑表达式>对。例如:

西单附近的餐厅

(lambda \$0 e (and (restaurant:t \$0) (zone:t \$0 西单:s)))

五道口有什么比较好吃的日本料理

---

(lambda \$0 e (and (restaurant:t \$0) (zone:t \$0 五道口:s) (label:t \$0 日本料理:s) (> (tasteScore:i \$0) 8:i))))

南京大牌档怎么样

(shopScore:i 南京大牌档:r)

南京大牌档有北京烤鸭吃吗

(hasCuisine:t 南京大牌档:r 北京烤鸭:c)

## 5.4 数据文件

数据及标注方案均在 **data** 文件夹中：

./data/types: 三元关系  $\langle e, p, e \rangle$  中实体 **e** 类型。

./data/relations: 关系 **p**。

./data/np\_lexicon: 数据中出现的名词实体及其对应逻辑表达式。

./data/train: 训练数据，问题及对应逻辑表达式。

# 六、实验

用 407 条标注数据进行十折交叉验证，结果如下：

Average precision : 0.935917709207183

Average recall : 0.8850000000000001

Average F1 : 0.9095032720998745

# 七、参考文献

- Steedman, M., & Baldridge, J. (2011). Combinatory categorial grammar. *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell.
- Zettlemoyer, L. S., & Collins, M. (2007, June). Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. In *EMNLP-CoNLL* (pp. 678-687).

- 
- Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., & Steedman, M. (2010, October). Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 1223-1233). Association for Computational Linguistics.
  - Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., & Steedman, M. (2011, July). Lexical generalization in CCG grammar induction for semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1512-1523). Association for Computational Linguistics.
  - Zettlemoyer, L. S., & Collins, M. (2012). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.
  - Artzi, Y., Lee, K., & Zettlemoyer, L. (2015). Broad-coverage ccg semantic parsing with amr. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.